VIT Vidyalankar Institute of Technology Accredited A+ by NAAC	**Department of Information Technology**

Semester	T.E. Semester VI
Subject	Soft Computing Lab
Subject Professor In-charge	Ms. Rasika Ransing
Laboratory	L05A

Student Name	Abhishek Pal
Roll Number	22101A0067

<table>
<tr><td>Experiment Number</td><td>11</td></tr>
<tr><td>Experiment Title</td><td>Implementation of logic gates using McCulloch Pitts Model</td></tr>
<tr><td>Code</td><td>

```
def threshold_function(p, theta):
  return 1 if p >= theta else 0

def mcculloch_pitts_AND(x1, x2):
  weights = [1, 1]
  threshold = 2
  p = (x1 * weights[0]) + (x2 * weights[1])
  y = threshold_function(p, threshold)
  print(f"Case: X1={x1}, X2={x2}")
  print(f"  Net Input P = ({x1}×1) + ({x2}×1) = {p}")
  print(f"  Threshold (θ) = {threshold}")
  print(f"  Condition: P ≥ θ → {p} ≥ {threshold} → {'True' if p >= threshold else
'False'}")
  print(f"  Output Y = {y}\n")

def mcculloch_pitts_OR(x1, x2):
  weights = [1, 1]
  threshold = 1
  p = (x1 * weights[0]) + (x2 * weights[1])
  y = threshold_function(p, threshold)
  print(f"Case: X1={x1}, X2={x2}")
  print(f"  Net Input P = ({x1}×1) + ({x2}×1) = {p}")
  print(f"  Threshold (θ) = {threshold}")
  print(f"  Condition: P ≥ θ → {p} ≥ {threshold} → {'True' if p >= threshold else
'False'}")
  print(f"  Output Y = {y}\n")
```

</td></tr>
</table>

```
def mcculloch_pitts_NOT(x):
    weight = -1
    threshold = 0
    p = (x * weight)
    y = threshold_function(p, threshold)
    print(f"Case: X={x}")
    print(f"  Net Input P = ({x}×{weight}) = {p}")
    print(f"  Threshold (θ) = {threshold}")
    print(f"  Condition: P ≤ θ → {p} ≤ {threshold} → {'True' if p <= threshold else
'False'}")
    print(f"  Output Y = {y}\n")

def mcculloch_pitts_XOR(x1, x2):
    weights = [1, 1, -2]  # Adjusted weights for XOR logic
    threshold = 1
    p = (x1 * weights[0]) + (x2 * weights[1]) + (x1 * x2 * weights[2])
    y = threshold_function(p, threshold)
    print(f"Case: X1={x1}, X2={x2}")
    print(f"  Net Input P = ({x1}×1) + ({x2}×1) + ({x1}×{x2}×-2) = {p}")
    print(f"  Threshold (θ) = {threshold}")
    print(f"  Condition: P ≥ θ → {p} ≥ {threshold} → {'True' if p >= threshold else
'False'}")
    print(f"  Output Y = {y}\n")

def mcculloch_pitts_NAND(x1, x2):
    weights = [-1, -1]  # Inverse of AND gate
    threshold = -1
    p = (x1 * weights[0]) + (x2 * weights[1])
    y = threshold_function(p, threshold)
    print(f"Case: X1={x1}, X2={x2}")
    print(f"  Net Input P = ({x1}×-1) + ({x2}×-1) = {p}")
    print(f"  Threshold (θ) = {threshold}")
    print(f"  Condition: P ≥ θ → {p} ≥ {threshold} → {'True' if p >= threshold else
'False'}")
    print(f"  Output Y = {y}\n")
def mcculloch_pitts_NOR(x1, x2):
    weights = [-1, -1]  # Inverse of OR gate
    threshold = 0  # Changed to 0 to correct the NOR logic
    p = (x1 * weights[0]) + (x2 * weights[1])
    y = threshold_function(p, threshold)
    print(f"Case: X1={x1}, X2={x2}")
    print(f"  Net Input P = ({x1}×-1) + ({x2}×-1) = {p}")
```

	`` print(f" Threshold (θ) = {threshold}") print(f" Condition: P ≥ θ → {p} ≥ {threshold} → {'True' if p >= threshold else 'False'}") print(f" Output Y = {y}\n") print("=== AND Gate ===\n") for x1 in [0, 1]: for x2 in [0, 1]: mcculloch_pitts_AND(x1, x2) print("=== OR Gate ===\n") for x1 in [0, 1]: for x2 in [0, 1]: mcculloch_pitts_OR(x1, x2) print("=== NOT Gate ===\n") for x in [0, 1]: mcculloch_pitts_NOT(x) print("=== XOR Gate ===\n") for x1 in [0, 1]: for x2 in [0, 1]: mcculloch_pitts_XOR(x1, x2) print("=== NAND Gate ===\n") for x1 in [0, 1]: for x2 in [0, 1]: mcculloch_pitts_NAND(x1, x2) print("=== NOR Gate ===\n") for x1 in [0, 1]: for x2 in [0, 1]: mcculloch_pitts_NOR(x1, x2) ```
Output	=== AND Gate === Case: X1=0, X2=0 Net Input P = (0×1) + (0×1) = 0 Threshold (θ) = 2 Condition: P ≥ θ → 0 ≥ 2 → False Output Y = 0 Case: X1=0, X2=1 Net Input P = (0×1) + (1×1) = 1

	Threshold (θ) = 2 Condition: P ≥ θ → 1 ≥ 2 → False Output Y = 0 Case: X1=1, X2=0 Net Input P = (1×1) + (0×1) = 1 Threshold (θ) = 2 Condition: P ≥ θ → 1 ≥ 2 → False Output Y = 0 Case: X1=1, X2=1 Net Input P = (1×1) + (1×1) = 2 Threshold (θ) = 2 Condition: P ≥ θ → 2 ≥ 2 → True Output Y = 1 === OR Gate === Case: X1=0, X2=0 Net Input P = (0×1) + (0×1) = 0 Threshold (θ) = 1 Condition: P ≥ θ → 0 ≥ 1 → False Output Y = 0 Case: X1=0, X2=1 Net Input P = (0×1) + (1×1) = 1 Threshold (θ) = 1 Condition: P ≥ θ → 1 ≥ 1 → True Output Y = 1 Case: X1=1, X2=0 Net Input P = (1×1) + (0×1) = 1 Threshold (θ) = 1 Condition: P ≥ θ → 1 ≥ 1 → True Output Y = 1 Case: X1=1, X2=1 Net Input P = (1×1) + (1×1) = 2 Threshold (θ) = 1 Condition: P ≥ θ → 2 ≥ 1 → True Output Y = 1 === NOT Gate === Case: X=0 Net Input P = (0×-1) = 0

Threshold (θ) = 0
Condition: P ≤ θ → 0 ≤ 0 → True
Output Y = 1

Case: X=1
Net Input P = (1×-1) = -1
Threshold (θ) = 0
Condition: P ≤ θ → -1 ≤ 0 → True
Output Y = 0

=== XOR Gate ===

Case: X1=0, X2=0
Net Input P = (0×1) + (0×1) + (0×0×-2) = 0
Threshold (θ) = 1
Condition: P ≥ θ → 0 ≥ 1 → False
Output Y = 0

Case: X1=0, X2=1
Net Input P = (0×1) + (1×1) + (0×1×-2) = 1
Threshold (θ) = 1
Condition: P ≥ θ → 1 ≥ 1 → True
Output Y = 1

Case: X1=1, X2=0
Net Input P = (1×1) + (0×1) + (1×0×-2) = 1
Threshold (θ) = 1
Condition: P ≥ θ → 1 ≥ 1 → True
Output Y = 1

Case: X1=1, X2=1
Net Input P = (1×1) + (1×1) + (1×1×-2) = 0
Threshold (θ) = 1
Condition: P ≥ θ → 0 ≥ 1 → False
Output Y = 0

=== NAND Gate ===

Case: X1=0, X2=0
Net Input P = (0×-1) + (0×-1) = 0
Threshold (θ) = -1
Condition: P ≥ θ → 0 ≥ -1 → True
Output Y = 1

Case: X1=0, X2=1
Net Input P = (0×-1) + (1×-1) = -1

	Threshold (θ) = -1 Condition: P ≥ θ → -1 ≥ -1 → True Output Y = 1 Case: X1=1, X2=0 Net Input P = (1×-1) + (0×-1) = -1 Threshold (θ) = -1 Condition: P ≥ θ → -1 ≥ -1 → True Output Y = 1 Case: X1=1, X2=1 Net Input P = (1×-1) + (1×-1) = -2 Threshold (θ) = -1 Condition: P ≥ θ → -2 ≥ -1 → False Output Y = 0 === NOR Gate === Case: X1=0, X2=0 Net Input P = (0×-1) + (0×-1) = 0 Threshold (θ) = 0 Condition: P ≥ θ → 0 ≥ 0 → True Output Y = 1 Case: X1=0, X2=1 Net Input P = (0×-1) + (1×-1) = -1 Threshold (θ) = 0 Condition: P ≥ θ → -1 ≥ 0 → False Output Y = 0 Case: X1=1, X2=0 Net Input P = (1×-1) + (0×-1) = -1 Threshold (θ) = 0 Condition: P ≥ θ → -1 ≥ 0 → False Output Y = 0 Case: X1=1, X2=1 Net Input P = (1×-1) + (1×-1) = -2 Threshold (θ) = 0 Condition: P ≥ θ → -2 ≥ 0 → False Output Y = 0
Conclusion	From the above test cases, we can see that the McCulloch-Pitts Model correctly implements the AND, OR, NOT, XOR, NAND and NOR gates, covering all possible input combinations and decision points. The results align with the expected truth tables for each gate, ensuring that the neuron model functions as intended. The

	threshold function effectively determines the output based on the net input (P) and predefined weights.